MuzzChat - Readme

Total Time Spent on the Project
I worked basically every evening throughout a whole week for a couple of hours (2-4 hours), with a few exceptions. I estimate that the total time spent on the project is between 18-22 hours.

iOS Target Version & Supported Devices
I chose iOS 15 as the target version because it meets Muzz's minimum requirement. The app supports iPhones and iPads in portrait mode, similar to the Muzz app.

Dependency Management
I chose CocoaPods as the dependency manager for the project. The Pods I selected align with the project requirements, which include:
- Realm (RealmSwift) for data storage.
- RxSwift (+RxCocoa) for data and control binding.
- Additionally, Kingfisher for quick asynchronous remote image downloading.

Extras
In addition to the required Chat screen, I decided to implement some additional components and functionalities such as:

- **Chat List Screen**: This screen demonstrates how navigation can be handled using simple coordinators (MVVM-C) and illustrates how SwiftUI and UIViewControllers can coexist within a single module. It also showcases aspects of protocol-oriented programming, as well as factory or dependency patterns, without relying on any external libraries.
- **Localization**: I localized the application to Arabic, allowing you to test it by running one of the two created shared schemes. The translations were prepared with the help of ChatGPT, and this feature demonstrates how the app's layout changes between LTR and RTL modes.
- **Preloaded Users and Messages**: I implemented preloaded users and messages, along with simulated responses in the chat. When a new message is sent, a mocked socket service first simulates the delivery of the message, followed by the message being marked as read. Additionally, there is a certain probability of automated follow-up responses being dropped once the message is read.
- **Extensions Organization** - I decided to place all extensions in a single Extensions.swift file, as opposed to creating separate files for each class or struct being extended, to maintain simplicity and organization within the project structure.

What Could Be Improved

- **Tests**: I added a few examples of XCTests to the project, but in an actual project, there would likely be more. Since the app doesn't perform any time-consuming processes, there was no need to implement "expectation/fulfillment" tests.
- **Error Handling**: Due to limited time resources, I couldn't implement full-scale error handling. As a result, you can find some parts where I force try Realm operations. However, I at least added a check to ensure that Realm is set up correctly when the app starts.

ChatGPT

I used ChatGPT to translate the application to Arabic, generate user names and pictures, mock messages more efficiently, and to compose and revise this README PDF file.

Feedback

Thank you for the opportunity to participate in this step of the recruitment process. I appreciated that this project was not just a typical "main list and detail module" with asynchronous data and image loading. The instructions were clear, and I found the project both enjoyable and challenging. I took my time to implement the requested details thoroughly, focusing on both the logic and the user interface.

Bartłomiej Wojdan

bwojdan@gmail.com